Adaptive Base Class Boost for Multi-class Classification

Ping Li
Department of Statistical Science
Faculty of Computing and Information Science
Cornell University
Ithaca, NY 14853
pingli@cornell.edu

First draft July 2008. Revised October 2008

Abstract

We[1] develop the concept of **ABC**-Boost (**A**daptive **B**ase **C**lass Boost) for multi-class classification and present **ABC**-MART, a concrete implementation of **ABC**-Boost. The original MART (**M**ultiple **A**dditive **R**egression **T**rees) algorithm has been very successful in large-scale applications. For binary classification, ABC-MART recovers MART. For multi-class classification, ABC-MART considerably improves MART, as evaluated on several public data sets.

1 Introduction

Classification is a basic task in machine learning. A training data set $\{y_i, \mathbf{X}_i\}_{i=1}^N$ consists of N feature vectors (samples) $\mathbf{X}_i$, and N class labels, y_i, $i = 1$ to N. Here $y_i \in \{0, 1, 2, ..., K-1\}$ and K is the number of classes. The task is to predict the class labels. This study focuses on multi-class classification ($K \geq 3$).

Many classification algorithms are based on **boosting**[7, 2, 3, 5, 4], which is regarded one of most significant breakthroughs in machine learning. MART[4] (**M**ultiple **A**dditive **R**egression **T**rees) is a successful boosting algorithm, especially for large-scale applications in industry practice. For example, the regression-based ranking method developed in Yahoo![1, 9] used an underlying learning algorithm based on MART. *McRank*[6], the classification-based ranking method, also used MART as the underlying learning procedure.

This study proposes **ABC**-Boost (**A**daptive **B**ase **C**lass Boost) for multi-class classification. We present **ABC**-MART, a concrete implementation of ABC-Boost. ABC-Boost is based on the following two key ideas:

1. For multi-class classification, popular loss functions for K classes usually assume a constraint[5, 4, 8] such that only the values for $K-1$ classes are needed. Therefore, we can choose a **base class** and derive algorithms only for $K-1$ classes.

2. At each boosting step, although the base class is not explicitly trained, it will implicitly benefit from the training on $K-1$ classes, due to the constraint. Thus, we **adaptively** choose the base class which has the "worst" performance.

The idea of assuming a constraint on the loss function and using a base class may not be at all surprising. For binary ($K = 2$) classification, a "sum-to-zero" constraint on the loss function is automatically considered so that we only need to train the algorithm for one (instead of $K = 2$) class. For multi-class ($K \geq 3$) classification, the sum-to-zero constraint on the loss function is also ubiquitously adopted[5, 4, 8]. In particular, the multi-class *Logitboost*[5] algorithm was derived by explicitly averaging over $K-1$ base classes.

[1]The first draft was submitted to IEEE ICDM on July 07, 2008. Although the submission was not accepted, the author appreciates two of the reviewers for their informative comments. In particular, the author appreciates that one of the reviewer considered this work "will be one of the state of the art." In this revision, we followed that reviewer's suggestion by adding more experiments.

The loss function adopted in our ABC-MART is the same as in MART[4] and *Logitboost*[5]. All three algorithms assume the "sum-to-zero" constraint. However, we obtain different first and second derivatives of the loss function, from MART[4] and *Logitboost*[5]. See Section 3 for details.

In terms of implementation, our proposed ABC-MART differs from the original MART algorithm only in a few lines of code. Since MART is known to be a successful algorithm, much of our work is devoted to the empirical comparisons of ABC-MART with MART. Our experiment results on publicly available data sets will demonstrate that ABC-MART could considerably improves MART. Also, ABC-MART reduces both the training and testing time by $1/K$, which may be quite beneficial when K is small.

We notice that data sets in industry applications are often quite large (e.g., several million samples[6]). Publicly available data sets (e.g., UCI repository), however, are mostly small. In our study, the *Covertype* data set from the UCI repository is reasonably large with 581,012 observations.

We first review the original MART algorithm and functional gradient boosting[4].

2 Review MART and Functional Gradient Boosting

The MART algorithm is the marriage of the functional gradient boosting and regression trees. Given the training data set $\{y_i, \mathbf{x}_i\}_{i=1}^N$ and a loss function L, [4] adopted a "greedy stagewise" approach to build an additive function $F^{(M)}$, which is a sum of M terms,

$$F^{(M)}(\mathbf{x}) = \sum_{m=1}^{M} \rho_m h(\mathbf{x}; \mathbf{a}_m), \tag{1}$$

such that, at each stage m, $m = 1$ to M,

$$\{\rho_m, \mathbf{a}_m\} = \underset{\rho, \mathbf{a}}{\text{argmin}} \sum_{i=1}^{N} L\left(y_i, F^{(m-1)}(\mathbf{x}_i; \mathbf{a}, \rho)\right). \tag{2}$$

Here the function $h(\mathbf{x}; \mathbf{a})$ is the "base learner" or "weak learner." In general, (2) is still a difficult optimization problem. [4] approximately conducted steepest descent in the function space, by solving a least square problem

$$\mathbf{a}_m = \underset{\mathbf{a}, \rho}{\text{argmin}} \sum_{i=1}^{N} \left[-g_m(\mathbf{x}_i) - \rho h(\mathbf{x}_i; \mathbf{a})\right]^2, \tag{3}$$

where

$$-g_m(\mathbf{x}_i) = -\left[\frac{\partial L\left(y_i, F(\mathbf{x}_i)\right)}{\partial F(\mathbf{x}_i)}\right]_{F((x))=F^{(m-1)}((x))} \tag{4}$$

is the negative gradient (the steepest descent direction) in the N-dimensional data space at $F^{(m-1)}(\mathbf{x})$. For obtaining another coefficient ρ_m, a line search is performed:

$$\rho_m = \underset{\rho}{\text{argmin}} \sum_{i=1}^{N} L\left(y_i, F^{(m-1)}(\mathbf{x}_i) + \rho h(\mathbf{x}_i; \mathbf{a}_m)\right). \tag{5}$$

A generic "gradient boosting" algorithm is described in Alg. 1, for any differentiable loss function L. For multi-class classification, [4] proposed MART, which implemented Line 5 in Alg. 1 by regression trees and Line 7 by a one-step Newton update within each terminal node of the trees.

2.1 The Loss Function and Multiple Logistic Probability Model in MART

For multi-class classification, MART adopted the following *negative multinomial log-likelihood loss*, which is also the loss function in *Logitboost*[5]:

$$L = \sum_{i=1}^{N} L_i = \sum_{i=1}^{N} \left\{ -\sum_{k=0}^{K-1} r_{i,k} \log p_{i,k} \right\} \tag{6}$$

Algorithm 1 A generic gradient boosting algorithm [4, Alg. 1].

```
1: $F_{\mathbf{x}} = \text{argmin}_\rho \sum_{i=1}^N L(y_i, \rho)$
2: For $m = 1$ to $M$ Do
3:     For $k = 0$ to $K-1$ Do
4:         $\tilde{y}_i = -\left[\frac{\partial L(y_i, F(\mathbf{x}_i))}{\partial F(\mathbf{x}_i)}\right]_{F((x))=F^{(m-1)}((x))}$, $i = 1$ to $N$.
5:         $\mathbf{a}_m = \text{argmin}_{\mathbf{a},\rho} \sum_{i=1}^N \left[\tilde{y}_i - \rho h(\mathbf{x}_i; \mathbf{a})\right]^2$
6:         $\rho_m = \text{argmin}_\rho \sum_{i=1}^N L\left(y_i, F^{(m-1)}(\mathbf{x}_i) + \rho h(\mathbf{x}_i; \mathbf{a}_m)\right)$
7:         $F_{\mathbf{x}} = F_{\mathbf{x}} + \rho_m h(\mathbf{x}; \mathbf{a}_m)$
8:     End
9: End
```

where $r_{i,k} = 1$ if $y_i = k$ and $r_{i,k} = 0$ otherwise. Apparently, $\sum_{k=0}^{K-1} r_{i,k} = 1$ for any i. Here $p_{i,k}$ is the probability that the ith observation belongs to class k:

$$p_{i,k} = \mathbf{Pr}\left(y_i = k | \mathbf{X}_i\right). \tag{7}$$

MART adopted the following logistic probability model[4]

$$F_{i,k} = \log p_{i,k} - \frac{1}{K}\sum_{s=0}^{K-1} \log p_{i,s}, \tag{8}$$

or equivalently[4],

$$p_{i,k} = \frac{e^{F_{i,k}}}{\sum_{s=0}^{K-1} e^{F_{i,s}}}. \tag{9}$$

Apparently, the model (8) implies $\sum_{k=0}^{K-1} F_{i,k} = 0$, the sum-to-zero constraint. In fact, since $\sum_{k=0}^{K-1} p_{i,k} = 1$, the model only has $K-1$ degrees of freedom. Some constraint on $F_{i,k}$ is necessary in order to obtain a unique solution. For binary ($K = 2$) classification, the sum-to-zero constraint is automatically enforced.

2.2 The Original MART Algorithm

Alg. 2 describes the MART algorithm for multi-class classification using *negative multinomial log-likelihood loss* (6) and multi-class logistic model (9).

Algorithm 2 MART[4, Alg. 6]. Note that in Line 6, the term $p_{i,k}(1-p_{i,k})$ replaces the equivalent term $|r_{i,k} - p_{i,k}|(1 - |r_{i,k} - p_{i,k}|)$ in [4, Alg. 6]

```
0: $r_{i,k} = 1$, if $y_i = k$, and $r_{i,k} = 0$ otherwise.
1: $F_{i,k} = 0$, $k = 0$ to $K-1$, $i = 1$ to $N$
2: For $m = 1$ to $M$ Do
3:     For $k = 0$ to $K-1$ Do
4:         $p_{i,k} = \exp(F_{i,k}) / \sum_{s=0}^{K-1} \exp(F_{i,s})$
5:         $\{R_{j,k,m}\}_{j=1}^J = J$-terminal node regression tree from $\{r_{i,k} - p_{i,k},\ \ \mathbf{x}_i\}_{i=1}^N$
6:         $\beta_{j,k,m} = \frac{K-1}{K}\frac{\sum_{\mathbf{x}_i \in R_{j,k,m}} r_{i,k} - p_{i,k}}{\sum_{\mathbf{x}_i \in R_{j,k,m}} (1-p_{i,k})p_{i,k}}$
7:         $F_{i,k} = F_{i,k} + \nu \sum_{j=1}^J \beta_{j,k,m} 1_{\mathbf{x}_i \in R_{j,k,m}}$
8:     End
9: End .
```

MART follows the generic paradigm of functional gradient boosting Alg. 1. At each stage m, MART solves the mean square problem (Line 5 in Alg. 1) by regression trees. MART builds K regression trees at each boosting step.

We elaborate in more detail on serval key components of MART.

2.2.1 The Functional Gradient (Pseudo Response)

MART performs gradient descent in the function space, using the gradient evaluated at the function values. For the ith data point, using the *negative multinomial log-likelihood loss* (6), i.e.,

$$L_i\left(F_{i,k}, y_i\right) = -\sum_{k=0}^{K-1} r_{i,k} \log p_{i,k} \tag{10}$$

and the probability model (9), [4] showed

$$\frac{\partial L_i}{\partial F_{i,k}} = -\left(r_{i,k} - p_{i,k}\right). \tag{11}$$

This explains the term $r_{i,k} - p_{i,k}$ in Line 5 of Alg. 2.

2.2.2 The Second Derivative and One-Step Newton Update

While only the first derivatives were used for building the structure of the trees, MART used the second derivatives to update the values of the terminal nodes by a one-step Newton procedure. [4] showed

$$\frac{\partial^2 L_i}{\partial F_{i,k}^2} = p_{i,k}\left(1 - p_{i,k}\right), \tag{12}$$

which explains Line 6 in Alg. 2.

(11) and (12) were also derived in the *Logitboost*[5]. However, in this paper we actually obtain different first and second derivatives.

2.2.3 K Trees for K Classes

For each i, there are K function values, $F_{i,k}$, $k = 0$ to $K-1$, and consequently K gradients. MART builds K regression tress at each boosting step. Apparently, the constraint $\sum_{k=0}^{K-1} F_{i,k} = 0$ will not hold.

Note that one can actually re-center the $F_{i,k}$ at the end of every boosting step so that the sum-to-zero constraint is satisfied after training. That is, one can insert a line

$$F_{i,k} \leftarrow F_{i,k} - \frac{1}{K}\sum_{s=0}^{K-1} F_{i,s}, \tag{13}$$

after Line 8 in Alg. 2 to make $\sum_{k=0}^{K-1} F_{i,k} = 0$. However, we observe that this re-centering step makes no difference in our experiments. We believe the sum-to-zero constraint should be enforced before the training instead of after the training, at every boosting step.

The case $K = 2$ is an exception. Because the two pseudo responses, $r_{i,k=0} - p_{i,k=0}$ and $r_{i,k=1} - p_{i,k=1}$ are identical with signs flipped (i.e., $\sum_{k=0}^{1} F_{i,k} = 0$ automatically holds), there is no need to build $K = 2$ tress. In fact, [4] presented the binary classification algorithm separately [4, Alg. 5], which is the same as [4, Alg. 6] by letting $K = 2$, although the presentations were somewhat different.

Line 6 of Alg. 2 contains a factor $\frac{K-1}{K}$. For binary classification, it is clear that the factor $\frac{1}{2}$ comes from the mathematical derivation. For $K \geq 3$, we believe this factor in a way approximates the constraint $\sum_{k=0}^{K-1} F_{i,k} = 0$. Because MART builds K trees while there are only $K-1$ degrees of freedom, a factor $\frac{K-1}{K}$ may help reduce the influence.

2.2.4 Three Main Parameters: J, ν, and M

Practitioners like MART partly because this great algorithm has only a few parameters, which are not very sensitive as long as they fall in some "reasonable" range. It is often fairly easy to identify the (close to) optimal parameters with limited tuning. This is a huge advantage, especially for large data sets.

The number of terminal nodes, J, determines the capacity of the base learner. MART suggested $J = 6$ often might be a good choice.

The shrinkage parameter, ν, should be large enough to make a sufficient progress at each boosting step and small enough to avoid over-fitting. Also, a very small ν may require a large number of boosting steps, which may be a practical concern for real-world applications if the training and/or testing is time-consuming. [4] suggested $\nu \leq 0.1$.

The number of boosting steps, M, in a sense is largely determined by the computing time one can afford. A commonly-regarded merit of boosting is that over-fitting can be largely avoided for reasonable J and ν and hence one might simply let M be as large as possible. However, for small data sets, the training loss (6) may reach the machine accuracy before M can be too large.

3 ABC-Boost and ABC-MART

We re-iterate the two key components in developing ABC-Boost (Adaptive Base Class Boost):

1. Using the popular constraint in the loss function, we can choose a **base class** and derive the boosting algorithm for only $K-1$ classes.

2. At each boosting step, we can **adaptively** choose the base class which has the "worst" performance.

ABC-MART is a concrete implementation of ABC-Boost by using the *negative multinomial log-likelihood loss* (6), the multi-class logistic model (9), and the paradigm of functional gradient boosting. Apparently, there are other possible implementations of ABC-Boost. For example, one can implement an "ABC-Logitboost." This study focuses on ABC-MART. Since MART has been proved to be successful in large-scale industry applications, demonstrating that ABC-MART may considerably improve MART will be appealing.

3.1 The Multi-class Logistic Model with a Fixed Base

We first derive some basic formulas needed for developing ABC-MART. Without loss of generality, we assume class 0 is the base class.

Lemma 1 provides the first and second derivatives of the class probabilities $p_{i,k}$ under the multi-class logistic model (9) and the sum-to-zero constraint $\sum_{k=0}^{K-1} F_{i,k} = 0$[5, 4, 8].

Lemma 1

$$\frac{\partial p_{i,k}}{\partial F_{i,k}} = p_{i,k}\left(1 + p_{i,0} - p_{i,k}\right), \quad k \neq 0 \tag{14}$$

$$\frac{\partial p_{i,k}}{\partial F_{i,s}} = p_{i,k}\left(p_{i,0} - p_{i,s}\right), \quad k \neq s \neq 0 \tag{15}$$

$$\frac{\partial p_{i,0}}{\partial F_{i,k}} = p_{i,0}\left(-1 + p_{i,0} - p_{i,k}\right), \quad k \neq 0 \tag{16}$$

Proof: *Note that* $F_{i,0} = -\sum_{k=1}^{K-1} F_{i,k}$. *Hence*

$$\begin{aligned}
p_{i,k} &= \frac{e^{F_{i,k}}}{\sum_{s=0}^{K-1} e^{F_{i,s}}} = \frac{e^{F_{i,k}}}{\sum_{s=1}^{K-1} e^{F_{i,s}} + e^{\sum_{s=1}^{K-1} -F_{i,s}}} \\
\frac{\partial p_{i,k}}{\partial F_{i,k}} &= \frac{e^{F_{i,k}}}{\sum_{s=0}^{K-1} e^{F_{i,s}}} - \frac{e^{F_{i,k}}\left(e^{F_{i,k}} - e^{-F_{i,0}}\right)}{\left(\sum_{s=0}^{K-1} e^{F_{i,s}}\right)^2} \\
&= p_{i,k}\left(1 + p_{i,0} - p_{i,k}\right).
\end{aligned}$$

The other derivatives can be obtained similarly. $\square$

Lemma 1 helps derive the derivatives of the loss function, presented in Lemma 2.

Lemma 2

$$\frac{\partial L_i}{\partial F_{i,k}} = (r_{i,0} - p_{i,0}) - (r_{i,k} - p_{i,k}), \tag{17}$$

$$\frac{\partial^2 L_i}{\partial F_{i,k}^2} = p_{i,0}(1-p_{i,0}) + p_{i,k}(1-p_{i,k}) + 2p_{i,0}p_{i,k}. \tag{18}$$

Proof:

$$L_i = -\sum_{s=1,s\neq k}^{K-1} r_{i,s}\log p_{i,s} - r_{i,k}\log p_{i,k} - r_{i,0}\log p_{i,0}.$$

It first derivative is

$$\frac{\partial L_i}{\partial F_{i,k}} = -\sum_{s=1,s\neq k}^{K-1} \frac{r_{i,s}}{p_{i,s}}\frac{\partial p_{i,s}}{F_{i,k}} - \frac{r_{i,k}}{p_{i,k}}\frac{\partial p_{i,k}}{F_{i,k}} - \frac{r_{i,0}}{p_{i,0}}\frac{\partial p_{i,0}}{F_{i,k}} \tag{19}$$

$$= \sum_{s=1,s\neq k}^{K-1} -r_{i,s}\left(p_{i,0} - p_{i,k}\right) - r_{i,k}\left(1 + p_{i,0} - p_{i,k}\right) - r_{i,0}\left(-1 + p_{i,0} - p_{i,k}\right)$$

$$= -\sum_{s=0}^{K-1} r_{i,s}\left(p_{i,0} - p_{i,k}\right) + r_{i,0} - r_{i,k}$$

$$= \left(r_{i,0} - p_{i,0}\right) - \left(r_{i,k} - p_{i,k}\right).$$

And the second derivative is

$$\frac{\partial^2 L_i}{\partial F_{i,k}^2} = -\frac{\partial p_{i,0}}{\partial F_{i,k}} + \frac{\partial p_{i,k}}{\partial F_{i,k}}$$

$$= -p_{i,0}\left(-1 + p_{i,0} - p_{i,k}\right) + p_{i,k}\left(1 + p_{i,0} - p_{i,k}\right)$$

$$= p_{i,0}(1-p_{i,0}) + p_{i,k}(1-p_{i,k}) + 2p_{i,0}p_{i,k}. \; \Box$$

Note the the first and second derivatives we derives differ from (11) and (12), which are the derivatives used in *Logitboost*[5] and MART[4].

3.2 ABC-MART

Alg. 3 provides the pseudo code for ABC-MART. Compared with MART (Alg. 2), we use different gradients to build the trees and different second derivatives to update the values of terminal nodes. In addition, there is a procedure (Line 10) for selecting the base class, denoted by b, which has the "worst" (i.e., largest) training loss (6). At each boosting step, we only need to build $K-1$ trees because the constraint $\sum_{k=0}^{K-1} F_{i,k} = 0$ is enforced.

3.3 ABC-MART Recovers MART when $K = 2$

Note that the factor $\frac{K-1}{K}$ does not appear in ABC-MART (Alg. 3). Interestingly, when $K = 2$, ABC-MART recovers MART.

For example, consider $K = 2$, $r_{i,0} = 1$, $r_{i,1} = 0$, then

$$\frac{\partial L_i}{\partial F_{i,1}} = (1 - p_{i,0}) - (0 - p_{i,1}) = 2p_{i,1},$$

$$\frac{\partial^2 L_i}{\partial F_{i,1}^2} = 4p_{i,0}p_{i,1}.$$

In other words, the first (second) derivative is twice (four times) of the first (second) derivative in MART. Using the one-step Newton update (Line 6 in Alg. 3), the factor $\frac{1}{2}$ (which appeared in MART) is recovered. Note that scaling the first derivatives does not affect the tree structures.

Algorithm 3 ABC-MART.

```
0: $r_{i,k} = 1$, if $y_i = k$, $r_{i,k} = 0$ otherwise. Choose a random base $b$
1: $F_{i,k} = 0,\ p_{i,k} = \frac{1}{K}, \quad k = 0$ to $K-1,\ i = 1$ to $N$
2: For $m = 1$ to $M$ Do
3:     For $k = 0$ to $K-1$, $k \neq b$, Do
4:         $\{R_{j,k,m}\}_{j=1}^{J} = J$-terminal node regression tree from $\{-(r_{i,b} - p_{i,b}) + (r_{i,k} - p_{i,k}),\ \mathbf{x}_i\}_{i=1}^{N}$
5:         $\beta_{j,k,m} = \frac{\sum_{\mathbf{x}_i \in R_{j,k,m}} -(r_{i,b}-p_{i,b})+(r_{i,k}-p_{i,k})}{\sum_{\mathbf{x}_i \in R_{j,k,m}} p_{i,b}(1-p_{i,b})+p_{i,k}(1-p_{i,k})+2p_{i,b}p_{i,k}}$
6:         $F_{i,k} = F_{i,k} + \nu \sum_{j=1}^{J} \beta_{j,k,m} 1_{\mathbf{x}_i \in R_{j,k,m}}$
7:     End
8:     $F_{i,b} = -\sum_{k\neq b} F_{i,k},\ i = 1$ to $N$.
9:     $p_{i,k} = \exp(F_{i,k}) / \sum_{s=0}^{K-1} \exp(F_{i,s}), \quad k = 0$ to $K-1,\ i = 1$ to $N$
10:    $b = \underset{k}{\text{argmax}}\ L^{(k)}$,  where  $L^{(k)} = \sum_{i=1}^{N} -\log(p_{i,k})\, 1_{y_i=k},\ k = 0, 1, ..., K-1$.
11: End
```

3.4 MART Approximately "Averages" All Base Classes when $K \geq 3$

In a sense, MART did consider the averaging affect from all $K-1$ base classes.

For the first derivatives, the following equality holds,

$$\sum_{b\neq k} \{-(r_{i,b} - p_{i,b}) + (r_{i,k} - p_{i,k})\} = K(r_{i,k} - p_{i,k}), \tag{20}$$

because

$$\begin{aligned}
&\sum_{b\neq k} \{-(r_{i,b} - p_{i,b}) + (r_{i,k} - p_{i,k})\} \\
=&-\sum_{b\neq k} r_{i,b} + \sum_{b\neq k} p_{i,b} + (K-1)(r_{i,k} - p_{i,k}) \\
=&-1 + r_{i,k} + 1 - p_{i,k} + (K-1)(r_{i,k} - p_{i,k}) \qquad \text{Recall : } \sum_{k=0}^{K-1} r_{i,k} = 1, \quad \sum_{k=0}^{K-1} p_{i,k} = 1 \\
=&K(r_{i,k} - p_{i,k}).
\end{aligned}$$

In other words, the gradient used in MART is the averaged gradient of ABC-MART.

Next, we can show that, for the second derivatives,

$$\sum_{b\neq k} \{(1-p_{i,b})p_{i,b} + (1-p_{i,k})p_{i,k} + 2p_{i,b}p_{i,k}\} \approx (K+2)(1-p_{i,k})p_{i,k}, \tag{21}$$

with equality holding only when $K = 2$, because

$$\begin{aligned}
&\sum_{b\neq k} \{(1-p_{i,b})p_{i,b} + (1-p_{i,k})p_{i,k} + 2p_{i,b}p_{i,k}\} \\
=&(K-1)(1-p_{i,k})p_{i,k} + 2p_{i,k}\sum_{b\neq k} p_{i,b} + \sum_{b\neq k}(1-p_{i,b})p_{i,b} \\
=&(K+1)(1-p_{i,k})p_{i,k} + \sum_{b\neq k} p_{i,b} - \sum_{b\neq k} p_{i,b}^2 \\
\approx&(K+1)(1-p_{i,k})p_{i,k} + \sum_{b\neq k} p_{i,b} - \left(\sum_{b\neq k} p_{i,b}\right)^2 \\
=&(K+1)(1-p_{i,k})p_{i,k} + (1-p_{i,k})p_{i,k} \\
=&(K+2)(r_{i,k} - p_{i,k}).
\end{aligned}$$

Thus, even the second derivative used in MART may be approximately viewed as the averaged second derivatives in ABC-MART. It appears the factor $\frac{K-1}{K}$ in MART may be reasonably replaced by $\frac{K}{K+2}$ (both equal $\frac{1}{2}$ when $K = 2$). This, of course, will not make a real difference because the constant (either $\frac{K-1}{K}$ or $\frac{K}{K+2}$) can be absorbed into the shrinkage factor ν.

4 Evaluations

The goal of the evaluation study is to compare ABC-MART (Alg. 3) with MART (Alg. 2) for multi-class classification. The experiments were conducted on one fairly large data set (*Covertype*) plus five small data sets (*Letter, Pendigits, Zipcode, Optdigits*, and *Isolet*); see Table 1.[2]

Table 1: Our experiments were based on six publicly available data sets. We randomly split the *Covertype* data set into a training set and test set. For all other data sets, we used the standard (default) training and test sets.

Data set	K	# training samples	# test samples	# features
Covertype	7	290506	290506	54
Letter	26	16000	4000	16
Pendigits	10	7494	3498	16
Zipcode	10	7291	2007	256
Optdigits	10	3823	1797	64
Isolet	26	6218	1559	617

In general, a comprehensive and fair comparison of two classification algorithms is a non-trivial task. In our case, however, the comparison task appears quite easy because ABC-MART and MART differ only in a few lines of code and their underlying base learners can be completely identical.

Ideally, we hope that ABC-MART will improve MART, for every set of reasonable parameters, J and ν. For the five small data sets, we experiment with every combination of number of terminal nodes J and shrinkage ν, where J and ν are chosen from

$$J \in \{4,\ 6,\ 8,\ 10,\ 12,\ 14,\ 16\}, \qquad \nu \in \{0.04,\ 0.06,\ 0.08,\ 0.1\}.$$

For the five small data sets, we let the number of boosting steps $M = 10000$. However, the experiments usually terminated well before $M = 10000$ because the machine accuracy was reached.

For the *Covertype* data set, since it is fairly large, we only experimented with $J = 6, 10, 20$ and $\nu = 0.04, 0.1$ and we limited M to be 16000, 11500, 6000, for $J = 6, 10, 20$, respectively.

4.1 Summary of Experiment Results

The test misclassification error is a direct measure of performance. MART and ABC-MART output K class probabilities $p_{i,k}$, $k = 0$ to $K-1$, for each observation i. To obtain the class labels, we adopt the commonly used rule

$$k' = \underset{k}{\text{argmax}}\ p_{i,k}. \tag{22}$$

We define R_{err}, the "relative improvement of test mis-classification errors," as

$$R_{err} = \frac{\text{mis-classification errors of MART} - \text{mis-classification errors of ABC-MART}}{\text{mis-classification errors of MART}} \times 100\ (\%). \tag{23}$$

Since we experimented with a series of parameters, J, ν, and M, we report the overall "best" (i.e., smallest) mis-classification errors in Table 2. Later, we will also report the more detailed mis-classification errors for every combination of J and ν, in Sections 4.2 to 4.7.

[2]All data sets are publicly available. The *Zipcode* data set is downloaded from `http://www-stat.stanford.edu/~tibs/ElemStatLearn/data.html` and all other data sets can be found from the UCI repository.

Table 2: Summary of test mis-classification errors.

Data set	MART	ABC-MART	R_{err} (%)	P-value
Covertype	11133	10203	8.4	4.4×10^{-11}
Letter	135	111	17.8	0.060
Pendigits	123	104	15.5	0.100
Zipcode	111	98	11.7	0.178
Optdigits	56	41	26.8	0.061
Isolet	84	69	17.9	0.107

Table 2 also provides the "P-value" of the one-sided t-test. The idea is to model the test error rate (i.e., the test mis-classification errors divided by the number of test samples) as a binomial probability and then conduct the t-test using the normal approximation of the difference of two binomial probabilities. We can see that for the *Covertype* data set, the improvement of ABC-MART over MART is highly significant (the P-value is nearly zero) under this test. For the five small data sets, the P-values are also reasonably small.

We shall mention that this t-test is very stringent when the error rate is small. In fact, we do not often see papers which calculated the P-values when comparing different classification algorithms.

Next, we present the detailed experiment results on the six data sets.

4.2 Experiments on the *Covertype* Data Set

Table 3 summarizes the test mis-classification errors along with the relative improvements (R_{err}), for every combination of $J \in \{6, 10, 20\}$ and $\nu \in \{0.04, 0.1\}$. For each J and ν, the smallest test mis-classification errors, separately for ABC-MART and MART, are the lowest points in the curves in Figure 2.

Table 3: ***Covertype***. The test mis-classification errors. The corresponding relative improvements (R_{err}, $\%$) of ABC-MART are included in parentheses.

(a) MART

	$\nu = 0.04$	$\nu = 0.1$
$J = 6$	20756	15959
$J = 10$	15862	12553
$J = 20$	13630	11133

(b) ABC-MART

	$\nu = 0.04$	$\nu = 0.1$
$J = 6$	17185 (17.2)	14230 (10.8)
$J = 10$	13064 (17.6)	11487 (8.5)
$J = 20$	11595 (14.9)	10203 (8.4)

To report the experiments in a more informative manner, Figure 1, Figure 2, and Figure 3, respectively, present the training loss, the test mis-classification errors, and the relative improvements, for the complete history of M boosting steps (iterations).

Figure 1 indicates that ABC-MART reduces the training loss (6) considerably and consistently faster than MART. Figure 2 demonstrates that ABC-MART exhibits considerably and consistently smaller test mis-classification errors than MART.

Figure 3 illustrates that the relative improvements of ABC-MART over MART, in terms of the mis-classification errors, may be considerably larger than the numbers reported in Table 3 if we stop the training earlier. This phenomenon may be quite beneficial for real-world large-scale applications when either the training or test time is part of the performance measure. For example, real-time applications (e.g., search engines) may not be able to afford to use models with a very large number of boosting steps.

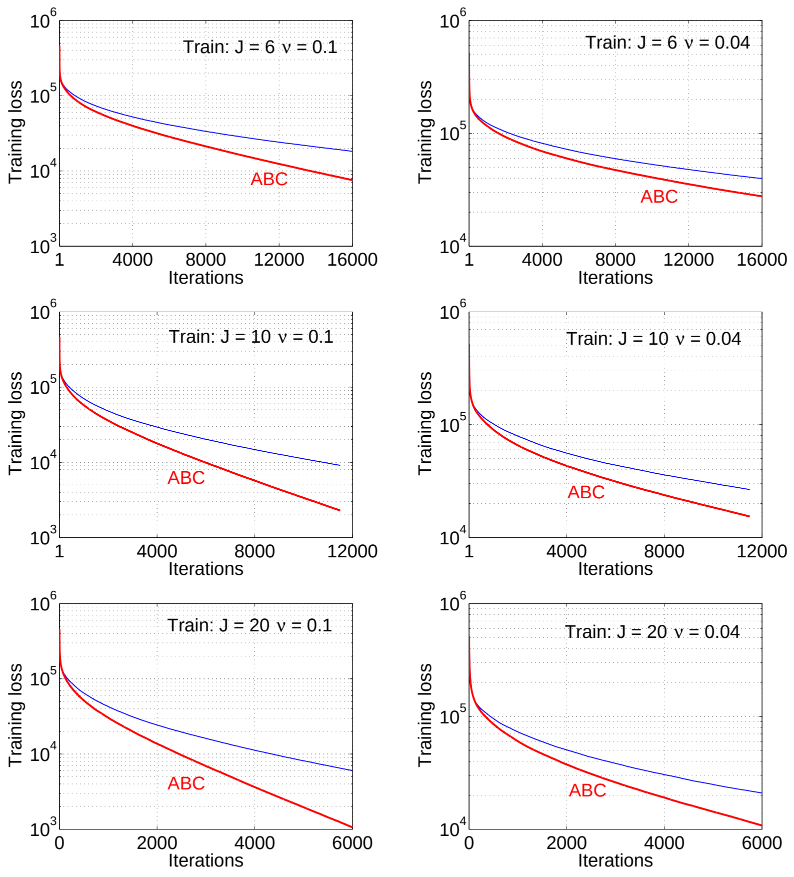

Figure 1: ***Covertype***. The training loss, i.e., (6). The curves labeled “ABC” correspond to ABC-MART.

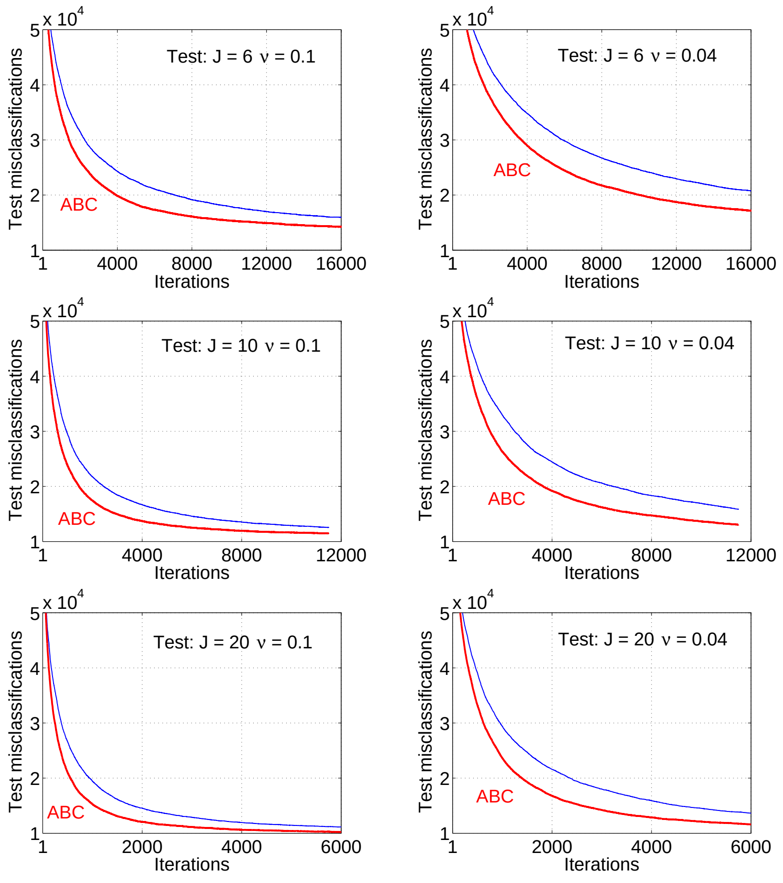

Figure 2: ***Covertype***. The test mis-classification errors.

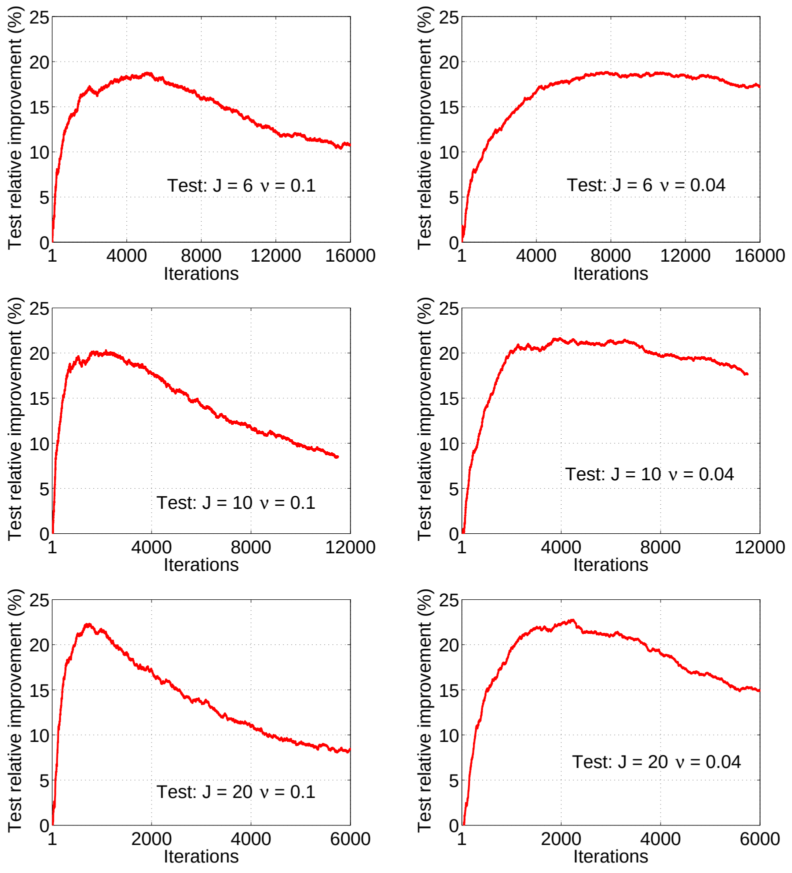

Figure 3: ***Covertype***. The relative improvements, R_{err} (%).

The next five subsections are respectively devoted to presenting the experiment results of five small data sets. The results exhibit similar characteristics across data sets.

4.3 Experiments on the *Letter* Data Set

Table 4 summarizes the test mis-classification errors along with the relative improvements for every combination of $J \in \{4,\ 6,\ 8,\ 10,\ 12,\ 14, 16\}$ and $\nu \in \{0.04,\ 0.06,\ 0.08,\ 0.1\}$.

Table 4: ***Letter***. The test mis-classification errors. The corresponding relative improvements (R_{err}, $\%$) of ABC-MART are included in parentheses.

(a) MART

	$\nu = 0.04$	$\nu = 0.06$	$\nu = 0.08$	$\nu = 0.1$
$J = 4$	176	178	177	173
$J = 6$	154	160	156	158
$J = 8$	151	145	151	154
$J = 10$	141	141	147	144
$J = 12$	150	144	144	140
$J = 14$	143	147	144	146
$J = 16$	138	147	142	135

(b) ABC-MART

	$\nu = 0.04$	$\nu = 0.06$	$\nu = 0.08$	$\nu = 0.1$
$J = 4$	155 (11.9)	143 (19.7)	148 (16.4)	144 (16.8)
$J = 6$	140 (9.1)	141 (11.9)	130 (16.7)	121 (23.4)
$J = 8$	132 (12.6)	126 (13.1)	124 (17.9)	117 (24.0)
$J = 10$	131 (7.1)	119 (15.6)	116 (21.1)	115 (20.1)
$J = 12$	125 (16.7)	124 (13.9)	119 (17.4)	119 (15.0)
$J = 14$	117 (18.2)	118 (19.7)	123 (14.6)	112 (23.3)
$J = 16$	117 (15.2)	113 (23.1)	113 (20.4)	111 (17.8)

Figure 4 again indicates that ABC-MART reduces the training loss (6) considerably and consistently faster than MART. Since this is a small data set, the training loss could approach zero even when the number of boosting steps is not too large.

Figure 5 again demonstrates that ABC-MART exhibits considerably and consistently smaller test mis-classification errors than MART.

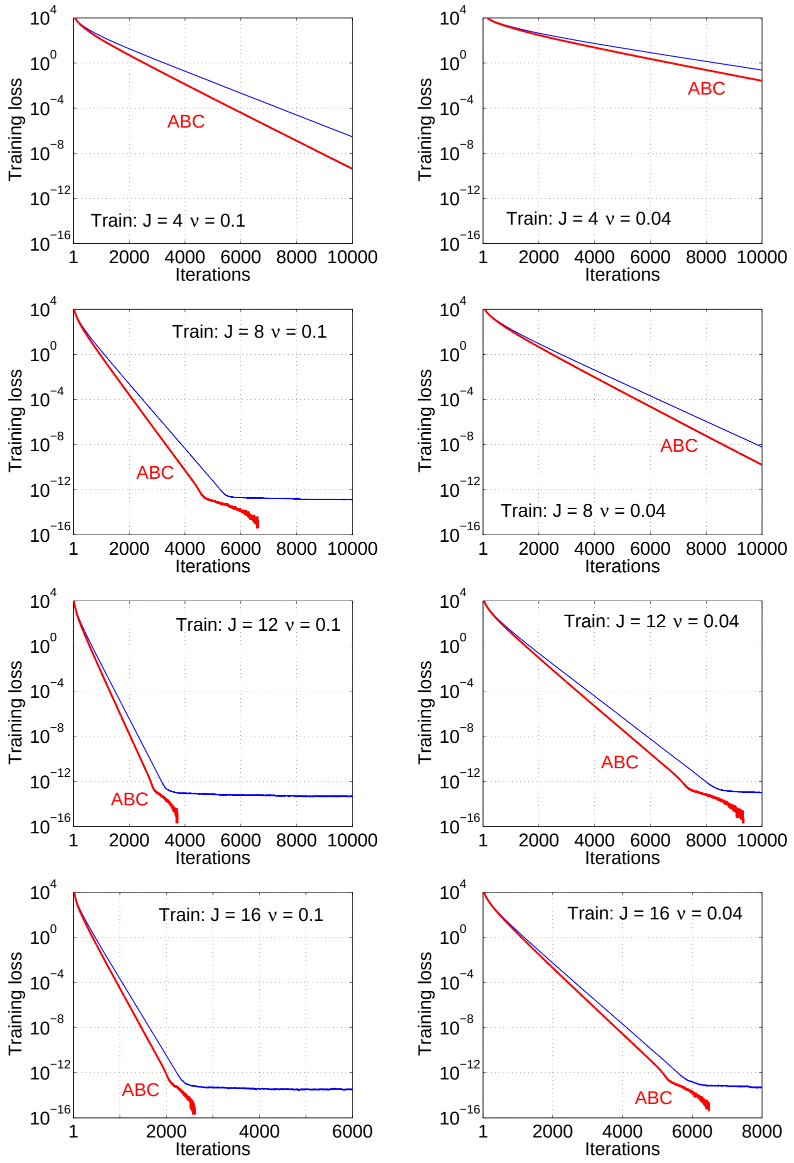

Figure 4: ***Letter***. The training loss, i.e., (6). The curves labeled "ABC" correspond to ABC-MART.

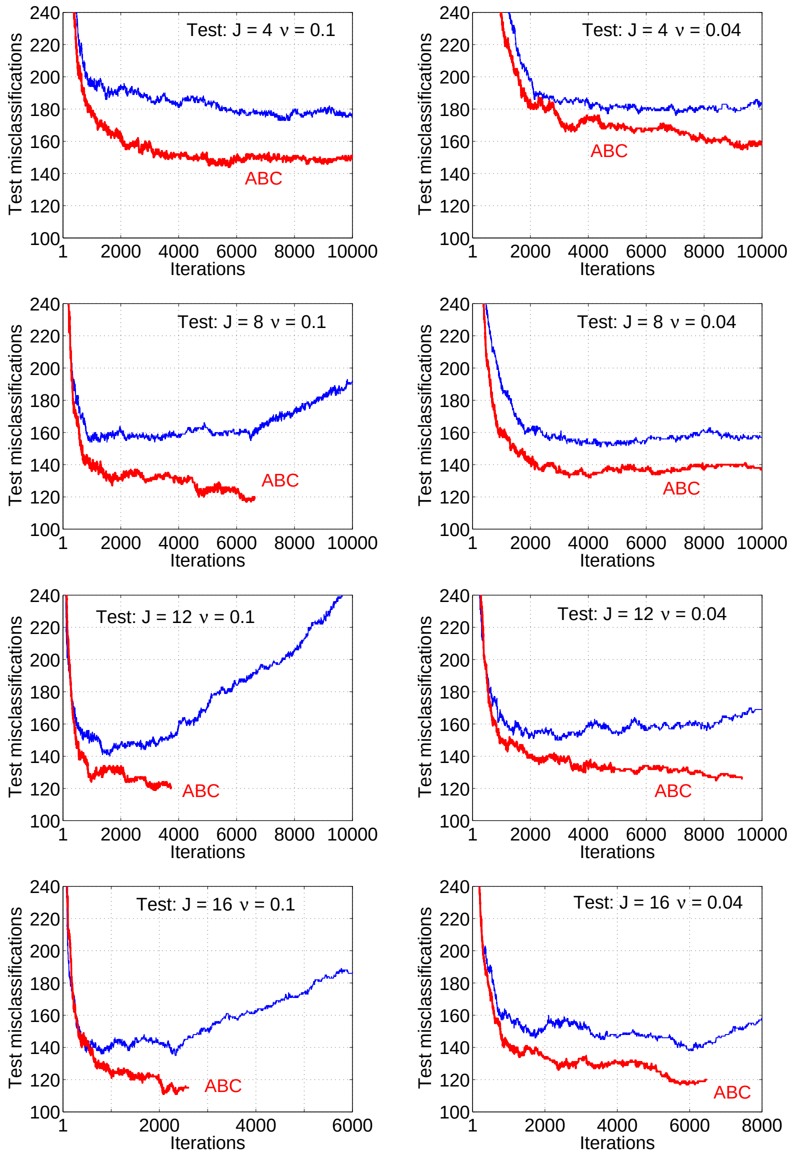

Figure 5: ***Letter***. The test mis-classification errors.

4.4 Experiments on the *Pendigits* Data Set

Table 5: ***Pendigits***. The test mis-classification errors. The corresponding relative improvements (R_{err}, $\%$) of ABC-MART are included in parentheses.

(a) MART

	$\nu = 0.04$	$\nu = 0.06$	$\nu = 0.08$	$\nu = 0.1$
$J = 4$	144	145	142	148
$J = 6$	137	134	136	135
$J = 8$	133	134	132	128
$J = 10$	123	131	127	130
$J = 12$	135	136	134	133
$J = 14$	129	131	130	133
$J = 16$	132	129	130	134

(b) ABC-MART

	$\nu = 0.04$	$\nu = 0.06$	$\nu = 0.08$	$\nu = 0.1$
$J = 4$	112 (22.2)	109 (24.8)	110 (22.5)	112 (24.3)
$J = 6$	114 (16.8)	114 (14.9)	111 (18.4)	109 (19.3)
$J = 8$	112 (15.8)	107 (20.1)	111 (15.9)	105 (18.0)
$J = 10$	105 (14.6)	105 (19.8)	107 (15.7)	109 (16.2)
$J = 12$	104 (23.0)	106 (22.1)	104 (22.4)	105 (21.1)
$J = 14$	107 (17.1)	106 (19.1)	104 (20.0)	107 (19.5)
$J = 16$	106 (19.7)	107 (17.1)	106 (18.5)	107 (20.1)

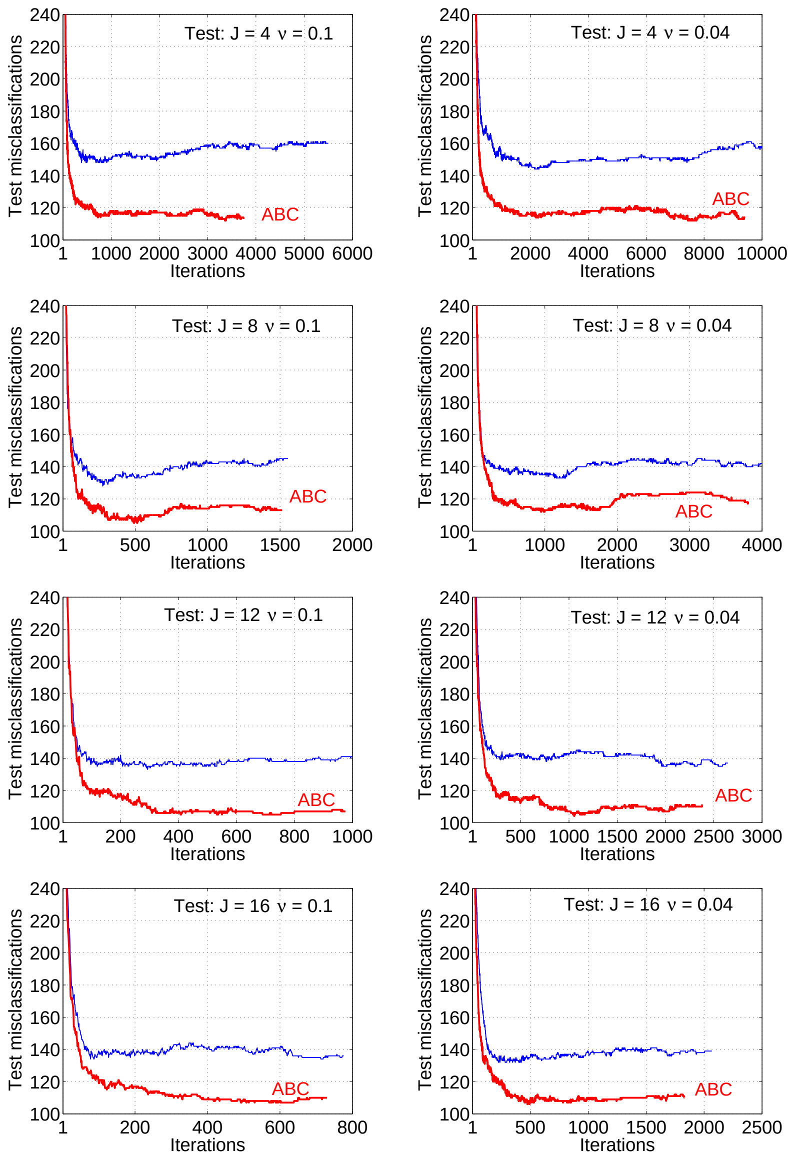

Figure 6: ***Pendigits***. The test mis-classification errors.

4.5 Experiments on the ***Zipcode*** Data Set

Table 6: ***Zipcode***. The test mis-classification errors.

(a) MART

	$\nu = 0.04$	$\nu = 0.06$	$\nu = 0.08$	$\nu = 0.1$
$J = 4$	130	126	129	127
$J = 6$	123	122	123	126
$J = 8$	120	122	121	123
$J = 10$	118	118	119	118
$J = 12$	117	117	116	118
$J = 14$	119	120	119	117
$J = 16$	118	111	116	116

(b) ABC-MART

	$\nu = 0.04$	$\nu = 0.06$	$\nu = 0.08$	$\nu = 0.1$
$J = 4$	116 (10.8)	114 (9.5)	116 (10.1)	111 (12.6)
$J = 6$	110 (10.6)	111 (9.0)	109 (11.4)	108 (14.3)
$J = 8$	109 (9.2)	102 (16.4)	111 (8.3)	106 (13.8)
$J = 10$	106 (10.2)	103 (12.7)	103 (13.4)	104 (11.9)
$J = 12$	105 (10.3)	102 (12.8)	103 (11.2)	98 (16.9)
$J = 14$	106 (10.9)	104 (13.3)	103 (13.4)	105 (10.3)
$J = 16$	104 (11.9)	103 (7.2)	102 (12.1)	104 (10.3)

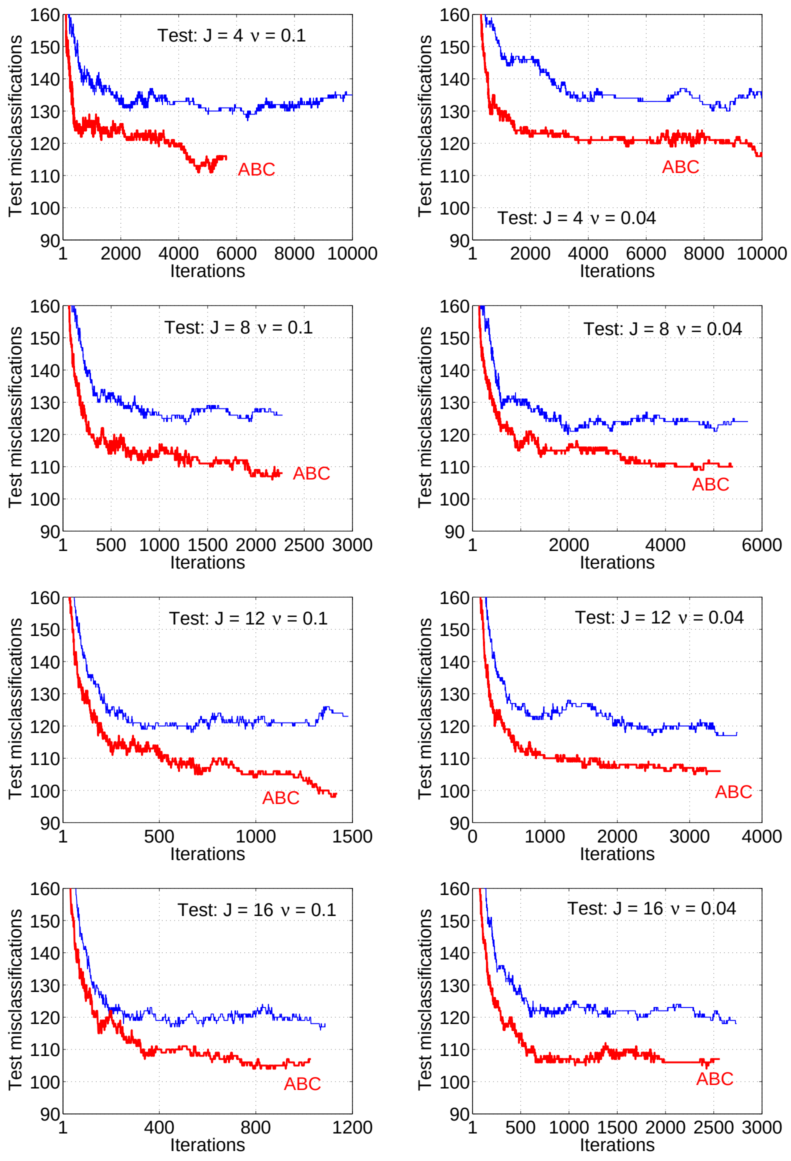

Figure 7: ***Zipcode***. The test mis-classification errors.

4.6 Experiments on the *Optdigits* Data Set

Table 7: ***Optdigits***. The test mis-classification errors

.

(a) MART

	$\nu = 0.04$	$\nu = 0.06$	$\nu = 0.08$	$\nu = 0.1$
$J = 4$	58	57	57	59
$J = 6$	60	57	59	57
$J = 8$	60	60	57	61
$J = 10$	59	59	58	59
$J = 12$	57	58	56	61
$J = 14$	57	59	58	56
$J = 16$	59	58	59	57

(b) ABC-MART

	$\nu = 0.04$	$\nu = 0.06$	$\nu = 0.08$	$\nu = 0.1$
$J = 4$	45 (22.4)	41 (28.1)	44 (22.8)	42 (28.8)
$J = 6$	47 (21.7)	48 (15.8)	47 (20.3)	45 (21.1)
$J = 8$	49 (18.3)	49 (18.3)	49 (14.0)	46 (24.6)
$J = 10$	55 (6.8)	54 (8.5)	50 (13.8)	49 (16.9)
$J = 12$	56 (1.8)	55 (5.2)	50 (10.7)	51 (16.4)
$J = 14$	55 (3.5)	53 (10.2)	50 (13.8)	47 (16.1)
$J = 16$	57 (3.4)	54 (6.9)	53 (10.2)	53 (7.0)

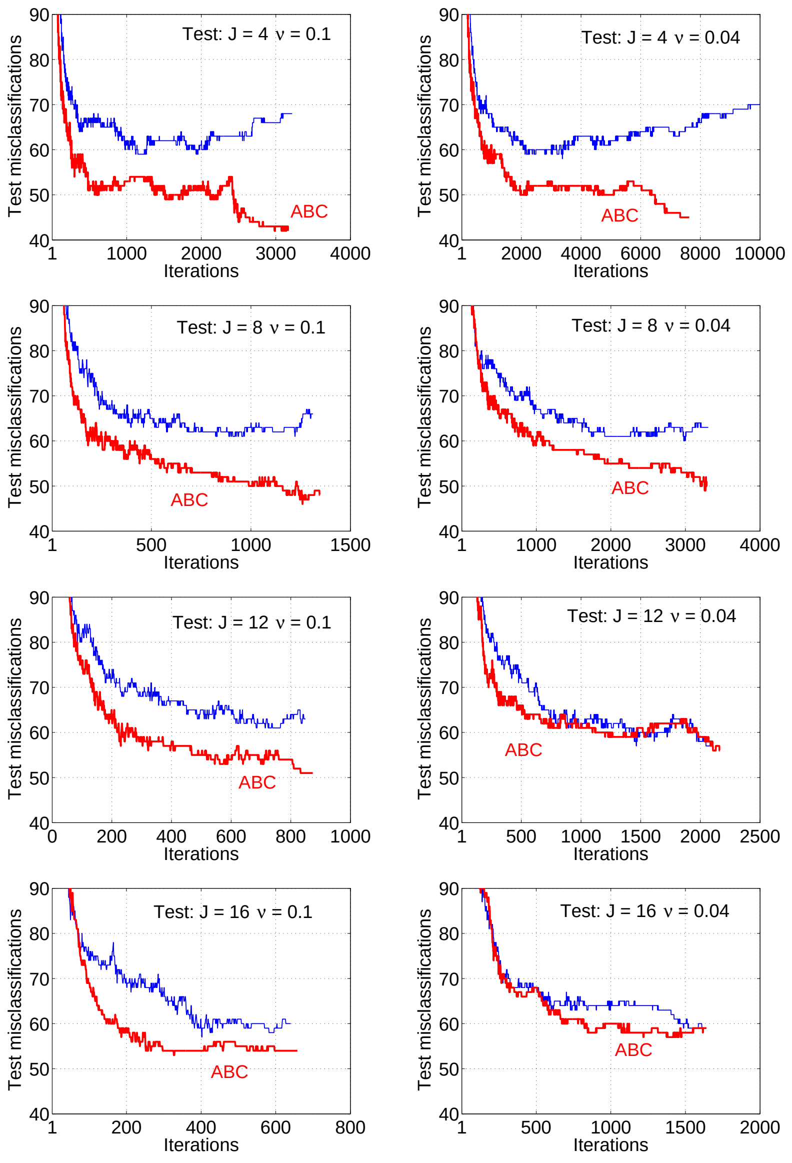

Figure 8: ***Optdigits***. The test mis-classification errors.

4.7 Experiments on the *Isolet* Data Set

Table 8: ***Isolet***. The test mis-classification errors.

(a) MART

	$\nu = 0.04$	$\nu = 0.06$	$\nu = 0.08$	$\nu = 0.1$
$J = 4$	84	84	85	84
$J = 6$	84	84	85	85
$J = 8$	87	89	86	85
$J = 10$	88	87	84	84
$J = 12$	88	91	85	86
$J = 14$	94	93	90	91
$J = 16$	93	91	90	88

(b) ABC-MART

	$\nu = 0.04$	$\nu = 0.06$	$\nu = 0.08$	$\nu = 0.1$
$J = 4$	74 (11.9)	74 (11.9)	74 (12.9)	72 (14.3)
$J = 6$	76 (9.5)	71 (15.5)	69 (18.8)	71 (16.5)
$J = 8$	72 (17.2)	74 (16.9)	73 (15.1)	73 (14.1)
$J = 10$	72 (18.2)	76 (12.6)	75 (10.7)	71 (15.5)
$J = 12$	78 (11.4)	74 (18.7)	75 (11.8)	72 (16.3)
$J = 14$	74 (21.3)	72 (22.6)	71 (21.1)	79 (13.2)
$J = 16$	79 (15.1)	75 (17.6)	76 (15.6)	79 (10.2)

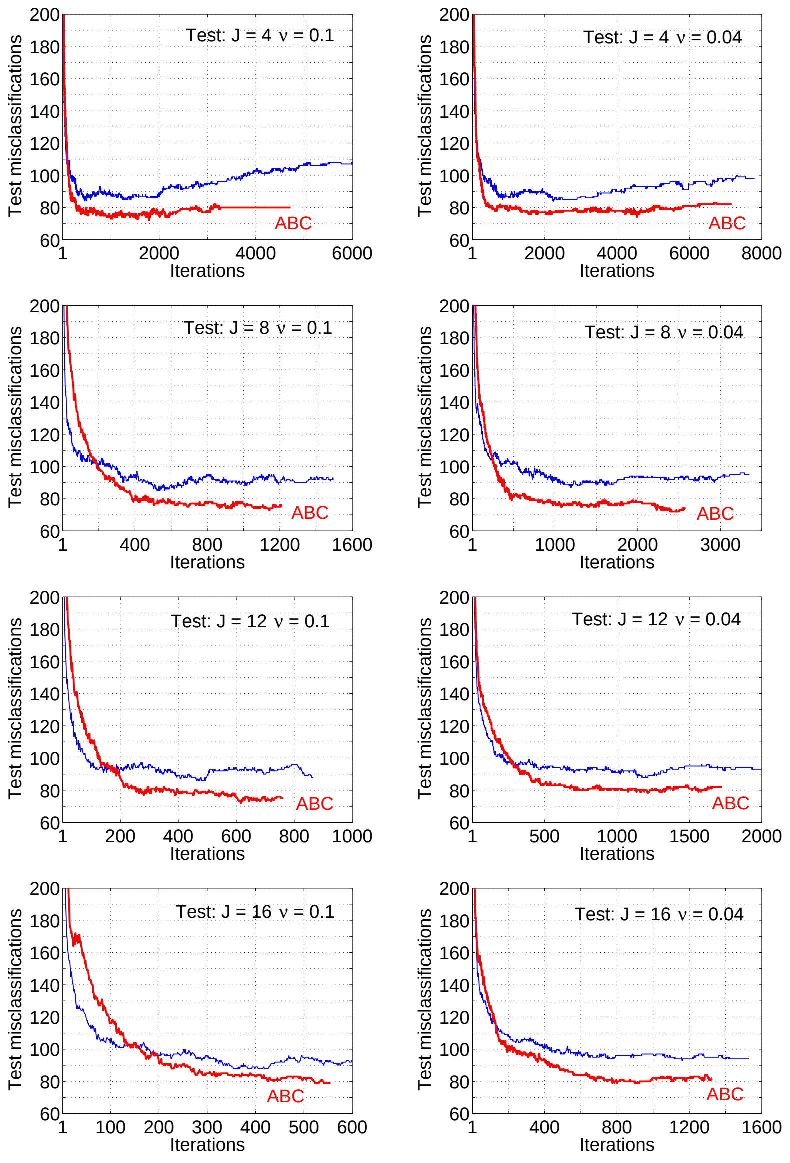

Figure 9: ***Isolet.*** The test mis-classification errors.

5 Discussion

Retrospectively, the ideas behind ABC-Boost and ABC-MART appear simple and intuitive. Our experiments in Section 4 have demonstrated the effectiveness of ABC-MART and its considerable improvements over MART.

The two key components of ABC-Boost are: (1) developing boosting algorithms by assuming one **base class**; (2) **adaptively** choosing the base class so that, at each boosting step, the "worst" class will be selected. We believe both components contribute critically to the good performance of ABC-MART.

Note that assuming the sum-to-zero constraint on the loss function and the base class is a ubiquitously adopted strategy[5, 4, 8]. Our contribution in this part is the different set of derivatives of the loss function (6), compared with the classical work[5, 4].

One may ask two questions. 1: Can we use the MART derivatives (11) and (12) and adaptively choose the base? 2: Can we use ABC-MART derivatives and a fixed base? Neither will achieve a good performance.

To demonstrate this point, we consider three alternative boosting algorithms and present their training and test results using the *Pendigits* data set ($K = 10$).

1. *MART derivatives + adaptively choosing the worst base*. It is the same as ABC-MART except it uses the derivatives of MART. In Figure 10, we label the corresponding curves by "Mb."

2. *ABC-MART derivatives + a fixed base class chosen according to MART training loss*. In the experiment with MART, we find "class 1" exhibits the overall largest training loss. Thus, we fix "class 1" as the base and re-train using the derivatives of ABC-MART. In Figure 10, we label the corresponding curves by "b1."

3. *ABC-MART derivatives + a fixed base class chosen according to MART test mis-classification errors*. In the experiment with MART, we find "class 7" exhibits the overall largest error. We fix "class 7" as the base and re-train using the derivatives of ABC-MART. In Figure 10, we label the corresponding curves by "b7."

Figure 10 demonstrate that none of the alternative boosting algorithms could outperform ABC-MART.

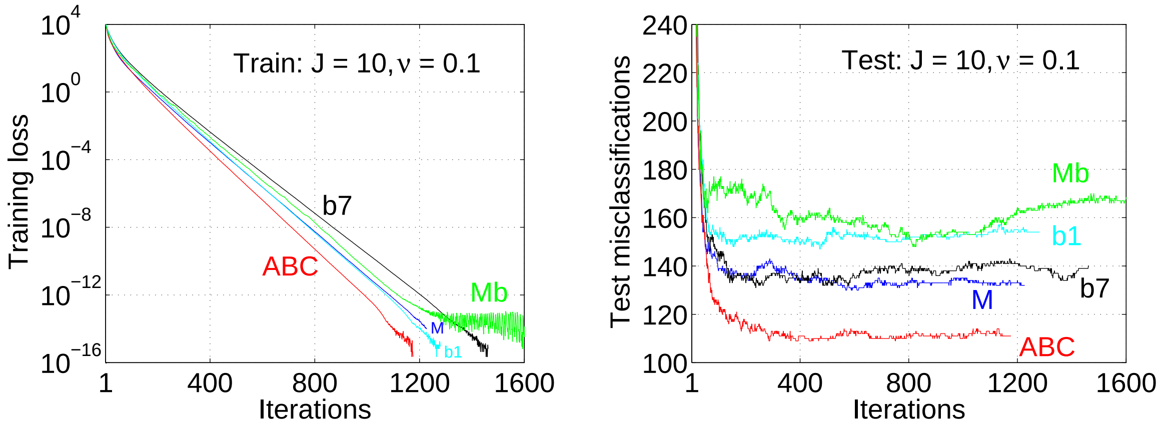

Figure 10: ***Pendigits***. The training loss and test mis-classification errors of three alternative boosting algorithms (labeled by "Mb", "b1", and "b7"), together with the results of ABC-MART and MART (labeled by "M").

6 Conclusion

We present the general idea of **ABC-Boost** and its concrete implementation named **ABC-MART**, for multi-class classification (with $K \geq 3$ classes). Two key components of ABC-Boost include: (1) By enforcing the (commonly used) constraint on the loss function, we can derive boosting algorithms for $K - 1$ class using a **base class**; (2) We **adaptively** choose the current "worst" class as the base class, at each boosting step. Both components are critical. Our experiments on a fairly large data set and five small data sets demonstrate that ABC-MART could considerably improves the original MART algorithm, which has already been highly successful in large-scale industry applications.

Acknowledgement

The author thanks Tong Zhang for providing valuable comments to this manuscript. In late June 2008, the author visited Stanford University and Google in Mountain View. During that visit, the author had the opportunity of presenting to Professor Friedman and Professor Hastie the algorithm and some experiment results of ABC-MART; and the author highly appreciates their comments and encouragement. Also, the author would like to express his gratitude to Phil Long (Google) and Cun-Hui Zhang for the helpful discussions of this work, and to Rich Caruana who suggested the author to test the algorithm on the *Covertype* data set.

References

[1] David Cossock and Tong Zhang. Subset ranking using regression. In *COLT*, pages 605–619, 2006.

[2] Yoav Freund. Boosting a weak learning algorithm by majority. *Inf. Comput.*, 121(2):256–285, 1995.

[3] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comput. Syst. Sci.*, 55(1):119–139, 1997.

[4] Jerome H. Friedman. Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29(5):1189–1232, 2001.

[5] Jerome H. Friedman, Trevor J. Hastie, and Robert Tibshirani. Additive logistic regression: a statistical view of boosting. *The Annals of Statistics*, 28(2):337–407, 2000.

[6] Ping Li, Christopher J.C. Burges, and Qiang Wu. Mcrank: Learning to rank using classification and gradient boosting. In *NIPS*, Vancouver, BC, Canada, 2008.

[7] Robert Schapire. The strength of weak learnability. *Machine Learning*, 5(2):197–227, 1990.

[8] Tong Zhang. Statistical analysis of some multi-category large margin classification methods. *Journal of Machine Learning Research*, 5:1225–1251, 2004.

[9] Zhaohui Zheng, Keke Chen, Gordon Sun, and Hongyuan Zha. A regression framework for learning ranking functions using relative relevance judgments. In *SIGIR*, pages 287–294, Amsterdam, The Netherlands, 2007.